AR-010-512

The Air Operations Simulation Centre Audio System

Reg Worl

DSTO-GD-0172

APPROVED FOR PUBLIC RELEASE

DEPARTMENT ◆ OF DEFENCE
DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

The Air Operations Simulation Centre Audio System

Reg Worl

Air Operations Division
Aeronautical and Maritime Research Laboratory

DSTO-GD-0172

ABSTRACT

The Air Operations Division at the Aeronautical and Maritime Research Laboratory has established the Air Operations Simulation Centre for conduct of manned aircraft flight simulations. An audio system has been developed to deliver sound effects during simulations and to provide audio communications between manned stations. This report describes the hardware and software components of the audio system.

RELEASE LIMITATION

Approved for public release

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

Published by

DSTO Aeronautical and Maritime Research Laboratory
PO Box 4331
Melbourne Victoria 3001

Telephone: (03) 9626 8111
Fax: (03) 9626 8999

AR No. AR-010-512
April 1998

APPROVED FOR PUBLIC RELEASE

The Air Operations Simulation Centre Audio System

Executive Summary

An Audio System for the Air Operations Simulation Centre (AOSC) has been assembled and is operational. The system provides audio communication facilities between manned stations and creates sound effects during aircraft flight simulations. Radio communication channels are modelled to create a realistic communications environment for pilot subjects participating in simulation experiments.

High-quality sound effects, used to create cockpit enunciators for example, can be reproduced from wavefiles played via an eight-channel audio sampler. Other sound effects requiring lower fidelity can be readily synthesised: cockpit alarms, wind and engine noises, morse code and navigation-beacon signals can be created to enhance the audio environment of the AOSC simulators.

The audio system is built around a digital multi-channel audio processor. This processor performs all audio switching, mixing and synthesis functions, and is easily reconfigured by loading new predefined models. To reduce reprogramming efforts, a flexible model for a general-purpose configuration of the audio processor was created that is suitable for most AOSC applications.

Software to interface the audio hardware to AOSC computers was developed. One program, for example, creates a graphical user interface which enables experiment operators to quickly adjust audio processor parameters controlling signal levels and frequencies of modelled radios.

The major system components- which comprise the audio sampler, the digital audio processor, and a high power audio amplifier- were assembled using commercial off-the-shelf components. All major components are mounted within one rack where a patch panel distributes signals to remote manned stations.

Contents

Table of Abbreviations

3D	Three Dimensional
AOD	Air Operations Division
AOSC	Air Operations Simulation Centre
ADF	Application Definition File
API	Application Programming Interface
ASTi	Advanced Simulation Technology inc.
AW2	Audio Works 2
CPU	Central Processing Unit
DACS	Digital Audio Communications System
dB	decibels
dc	direct current
DIS	Distributed Interactive Simulation
DSP	Digital Signal Processor
DSTO	Defence Science and Technology Organisation
EOS	Experiment Operator Station
GUI	Graphical User Interface
MIDI	Musical Instrument Digital Interface
PC	Personal Computer
PDD	Partial Dome Display
SGI	Silicon Graphics Inc.
SCSI	Small Computer System Interface
T-R-S	Tip-Ring-Sleeve
UDP	Universal Data Packet
VOX	Voice Operated Switch

1. Introduction

A growing area of research at DSTO's Air Operations Division (AOD) is in the use of manned simulations to design and assess aircraft systems, and to model and analyse airborne strategies and tactics. This research is carried out in the Air Operations Simulation Centre (AOSC) [1][2] which provides a flexible real-time simulation environment. Using reconfigurable displays, cockpits and computer systems, experiments can be undertaken to investigate the performance of aircrews and aircraft systems.

Audible warnings, background sounds and pilot-communication systems are an important element of any simulated aircraft environment. Sound requirements can vary widely between experiments so a versatile and reconfigurable audio system is required. Whether an experiment investigates the benefits of a three-dimensional audio target designator to an individual pilot or requires simulated radio-communications among several pilots for coordinated actions, the one audio system should accommodate all requirements.

This report presents the details of the AOSC audio system. It outlines the basic system architecture, describes the features of the main components, and discusses the distribution of audio throughout the AOSC.

2. AOSC Audio Requirements

The requirements of the AOSC Simulation System where initially defined in early work undertaken with assistance from consultants supplied by Adacel Pty Ltd. The documents 'System Segment Specification for the Phase 1 Integration of the Air Operations Simulation Centre' [3], and 'System Segment Design Document for the Phase 1 Integration of the Air Operations Simulation Centre' [4] were produced by Adacel Pty Ltd as a result of this work. These documents formed the basis of the AOSC development.

In Reference 4 the basic audio system was defined in terms of audio facilities required by the researchers and operators conducting the experiments as well as those required by the experiment subjects participating as pilots. The key components necessary to implement the required functions of the audio system were identified, and some of the necessary control interfaces were briefly listed in Data Flow Diagrams.

Three major components were specified at the heart of the audio system: a sound engine which creates the required sound effects from prerecorded sound wavefiles, a digital multi-channel mixer which provides sound mixing and implements an audio switcher for radio or intercom communications, and the Crystal Rivers Inc.

Convolvotron® convolution engine which filters audio to provide sounds positioned in three dimensional (3D) space.

In most cases off-the-shelf equipment was purchased to assemble the necessary components of the audio system as defined in Reference 4. Some additional features provided with this equipment have made the audio system more versatile than originally anticipated. Furthermore, in recent 3D-audio experiments, the requirements for replaying large numbers of pre-recorded sound files has seen the audio facilities on the Silicon Graphics Inc. (SGI) Onyx computer being developed to complement the audio system sound engine.

Although developed to comply with the requirements of Reference 4, the present audio system has grown due to AOSC extensions and now includes audio facilities for three additional simulator stations. It is not intended to reproduce here details or summaries of the requirements for headsets, microphones, or speakers at the simulator or control stations defined in Reference 4, however the reader may refer to Appendix A for a list of the resources which have been provided for the AOSC Audio System.

3. Equipment Components of the AOSC Audio System

3.1 The Sound Engine

A facility for the generation of sound effects in AOSC applications has been purchased from Paradigm Simulation Inc and consists of the *AudioWorks2* (AW2) software and an E-mu Systems Inc. EIIIxp digital sampler. This facility provides up to eight audio channels and sixteen stereo voices for rendering high fidelity sounds under real-time computer control. Output channels are assigned in pairs providing stereo sound effects for up to four AOSC experiment subjects.

The E-mu Systems Inc. EIIIxp sound engine contains front-panel switches which allow limited manual control of the sampler functions. Details on the operation of these controls are provided in Reference 5. A Musical Instrument Digital Interface (MIDI) is also available which together with the EMAGIC LOG2 MIDI buffer enables control commands to be transmitted from suitable serial ports on SGI or Apple Macintosh computers. However, in AOSC applications using *AW2*, a Small Computer System Interface (SCSI) port which is available on some of the SGI computers is used[1]. This high-speed interface is the prime method of down-loading sound wavefiles into the sound engine. When creating sound effects for AOSC simulations, the sound states, volumes and pitches are all controlled via this interface.

[1] At the time of writing, the SCSI port on the AOSC Onyx machine named '*Elric*' was being used for AW2 applications.

The AW2 software manages the control of all sound engine functions. Apart from maintaining the SCSI link to the sound engine and supplying output connections for the audio channels, the user never needs to operate the sound engine directly.

3.2 The Digital Multi-channel Mixer.

The digital multi-channel mixer functions detailed in Reference 4 are implemented using the Advanced Simulation Technologies Inc. (ASTi) Digital Audio Communication System (DACS). This system is built around a Personal Computer (PC) containing a 80486 central processing unit (CPU) and proprietary Digital Signal Processor (DSP) cards to provide twenty-four sampled-audio input and output channels. The software supplied with the system enables it to be configured to provide the signal mixing and routing required within the AOSC audio system and to simulate radio communications between active simulation participants. See Reference 6 for specifications and details on the use of the DACS equipment.

Programming the DACS to create specific configurations required in individual experiments is difficult without training and experience. To minimise programming efforts, a general-purpose system configuration has been created which is suitable for many AOSC applications. This configuration can be tuned by operators at remote terminals or workstations using the *mixer_gui* control program. By adjusting slider controls in this graphical user interface (GUI), parameters in the DACS configuration can be instantly updated to alter gains in selected signal paths. This enables unwanted signal paths in the default configuration to be removed by setting signal gains to zero.

The initial AOSC mixer configuration created for the DACS is illustrated in Figure 1. This configuration provides communications between six users and includes limited facilities for recording and monitoring playback. Note the numbered crosspoints shown in the diagram which represent the adjustable-gain signal paths leading to the mixer outputs. These are the controls which may be tuned by experiment operators using the *mixer_gui* windows.

The numbers attached to the gain controls in Figure 1 refer to the address of the relevant gain-control parameter as a byte-offset within the Universal Data Packet (UDP) transmitted by the *mixer* program to the DACS. These addresses also appear in the *mixer.cfg* configuration file where the corresponding slider controls are defined. The *mixer* program places slider values from the *mixer_gui* controls at the destination offset addresses obtained from this file.

The microphone inputs shown are applied to individual voice-operated-switch(VOX)-controlled radio models. When a microphone input level exceeds the VOX threshold, the attached radio transmits to all stations tuned to its transmit frequency. Radio outputs can then be heard via user headsets or user speakers where available.

The DACS provides the capability to synthesise sounds and to also playback wavefile recordings stored on its internal hard disk. The synthesised sounds may range from

simple modulated tones representing cockpit alarms or radio navigation beacons, to complex waveforms representing aircraft engines, propellers, or helicopter blades. Most AOSC applications will be developed using these features.

The DACS was developed specifically for use in simulators, and within the Australian Defence Organisation similar units can be found in use at Her Majesties Australian Ship *Watson* and in the Royal Australian Air Force F-111 simulator developed by Wormald Technology. These units are compatible with Distributed Interactive Simulation (DIS) protocols thereby providing communications with remote simulators using simulated radio transmissions.

Two ethernet interfaces are provided in the DACS to enable the use of separate domains for DIS traffic and local control of model/mixer parameters. When conducting DIS exercises, it is desirable to restrict the often heavy DIS traffic to an isolated domain to reduce congestion on local area networks supporting users and their applications.

3.3 The Convolution Engine

The convolution engine provides a facility for positioning sounds in 3D space. By using DSP filters to implement Head-Related Transfer Functions (HRTFs), the convolution engine filters monophonic signals so they may be heard through stereo headphones as external sounds emanating from different locations around the listener. The Convolvotron, manufactured by Crystal Rivers Inc., has been acquired for this purpose, and it processes up to four separate monophonic signals into a single stereo output.

The Convolvotron is built into a stand-alone PC containing a 80486-based CPU and a DSP engine. It is housed in a high-tower PC cabinet and is located adjacent to the AOSC audio rack. It has been wired to connectors on the front panel of the audio rack but has no mixer channels allocated for its use. When spatially positioned sounds are required, signal sources can be manually patched through the Convolvotron.

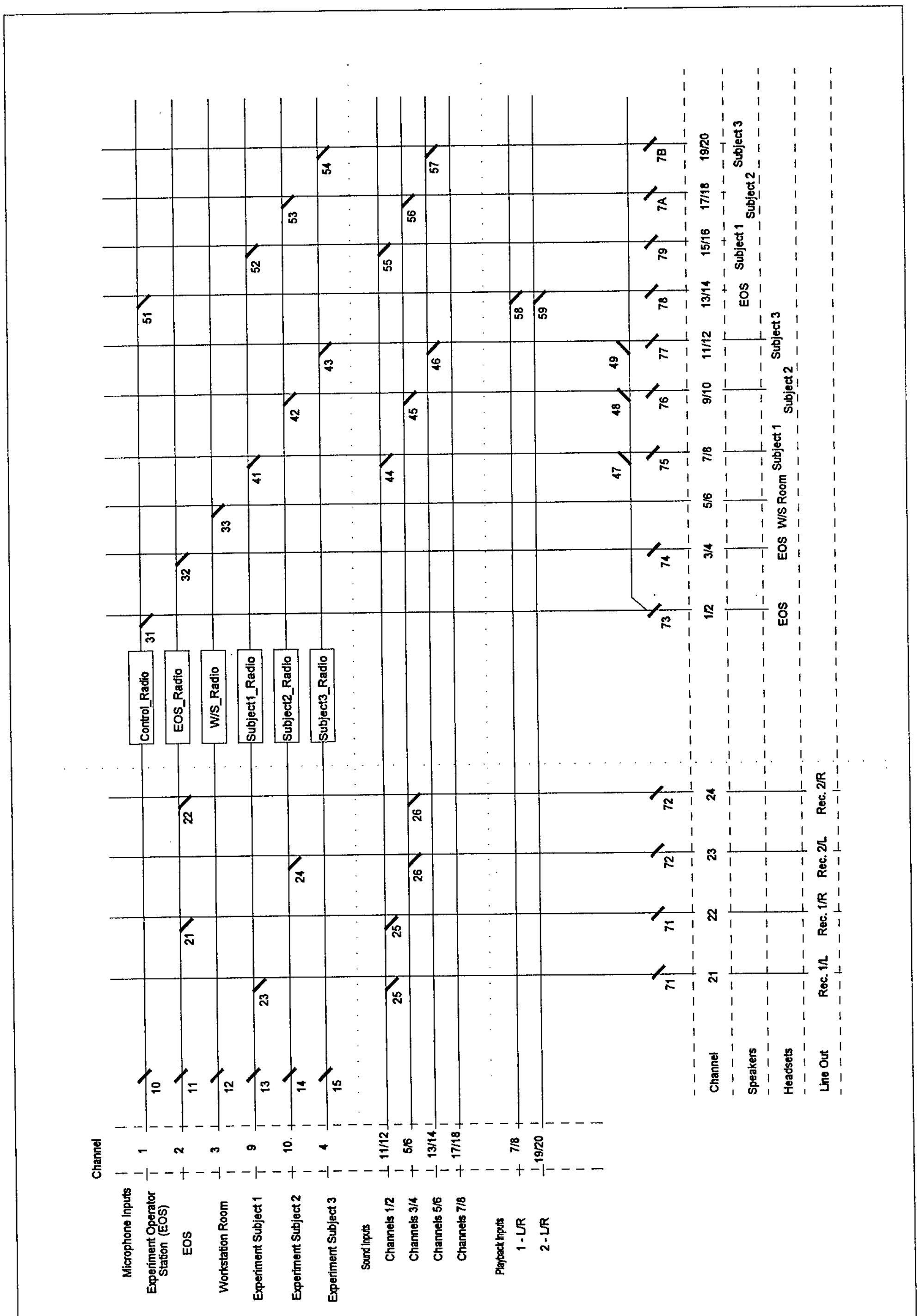

Figure 1 Initial configuration of the Digital Mixer.

Currently it is believed that high audio sampling rates in the order of 100,000 samples per second may be required to enable the position of spatialised sounds to be accurately perceived. Since the DACS digital mixer operates at a maximum 44,100 samples per second, it may be desirable in 3D sound experiments for the Convolvotron signals to bypass the DACS completely. The use of a separate analog mixer should therefore be considered if mixing functions are required with spatialised sound signals.

Superseded by the Acoustatron II, the Convolvotron performance is limited when compared to the superior effects achieved elsewhere in AOD using the Tucker-Davis Audio Processor. Although not suited to research studies into the effects of spatialised sounds, the Convolvotron remains useful in positioning sounds for demonstration purposes.

3.4 Standard SGI Audio Features

Most SGI workstations include an audio port which provides an input for electret microphones, an output for stereo headphones, and inputs and outputs for stereo line-level analog and stereo digital signals. The AOSC Onyx2 machine *'Gkar'* and the Onyx machine *'Elric'* with the optional audio port installed provide similar facilities.

The SGI audio ports are fully supported by the *AW2* software: all sound recording, editing and playback can be achieved by using an audio port with *AW2*. As an alternative to the sound engine, the SGI audio ports can also be used to create sound effects for simulations requiring a single stereo sound source.

Since the E-mu Systems Inc. EIIIxp sound engine lacks recording facilities, new sound wavefile recordings must be made on other machines then transferred to the sound engine. New sounds can be conveniently recorded using the line-level analog or digital inputs on any SGI audio port. The recorded wavefiles may then be transferred across the computer network to the sound engine host which down-loads wavefiles into the sound engine.

4. Supporting Software

Various software packages have been procured or developed to support the hardware components discussed in the previous section. The main packages of interest are:

- the AW2 software supplied for the E-mu Systems Inc. EIIIxp sound engine,
- the *sound_fx* program written to interface AOSC applications with the AW2 *AWD* daemon controlling the sound engine,

- the *mixer* and *mixer_gui* programs written to provide operators with a graphical interface for controlling digital mixer parameters, and
- a collection of audio media tools supplied with the SGI workstations.

These items are described below in greater detail. While reading the following sections, it may be useful to refer to the context diagram shown in Figure 2 which illustrates the relationships between the various software packages, their outputs, and the hardware which creates the audio output.

4.1 The AudioWorks2 Software

The *AW2* software supplied with the sound engine comprises four major items: *Audition* , a soundfile editor that includes loop-setting, fade-in, fade-out, and graphic equalisation functions for processing sound wavefiles; *AWLynx*, a configuration editor used to assign sounds to sound-engine audio channels; *AWD*, a daemon process that manages the control interface with the sound engine; and the application program interface (API) functions provided in several *AW2* libraries.

The *AW2* software processes sound wavefiles recorded in the AIFF or AIFC formats. These are the common sound-file formats used by SGI machines. Sound files supplied in other formats can be used by converting the formats using the *soundfiler* program supplied with the SGI system software.

The Audition soundfile editor creates AIFF or AIFC-format files from direct recordings of the microphone, line-level analog, or digital inputs on the SGI hardware. Standard sound-editing functions are provided together with a spectrum analyser and a graphical equaliser for basic tone adjustments. The standard sound-editing functions are similar to those of the *soundeditor* supplied with the SGI systems so the user may use either editor for similar results.

The sound environment required in a simulation must be defined prior to running any *AW2* application. This environment consists of *scenes* which group the sounds played through particular sound engine channels. In *scenes* using multi-channel audio outputs, the movement of listener earpoints and sound sources is simulated by adjusting sound balances between channels. The dynamic sound sources and listeners are created by attaching them to moving *player* entities.

The configuration editor '*AWLynx*' is provided with *AW2* to create specific Application Definition Files (ADFs) which define the relevant environment parameters. These files provide the locations of *players* and sound sources, lists of the required *scenes* and soundfiles, and the allocation of audio channels. When loaded by application programs, these files configure the *AW2* hardware to provide the necessary sound channels and cause the required sound wavefiles to be down-loaded into the sound engine.

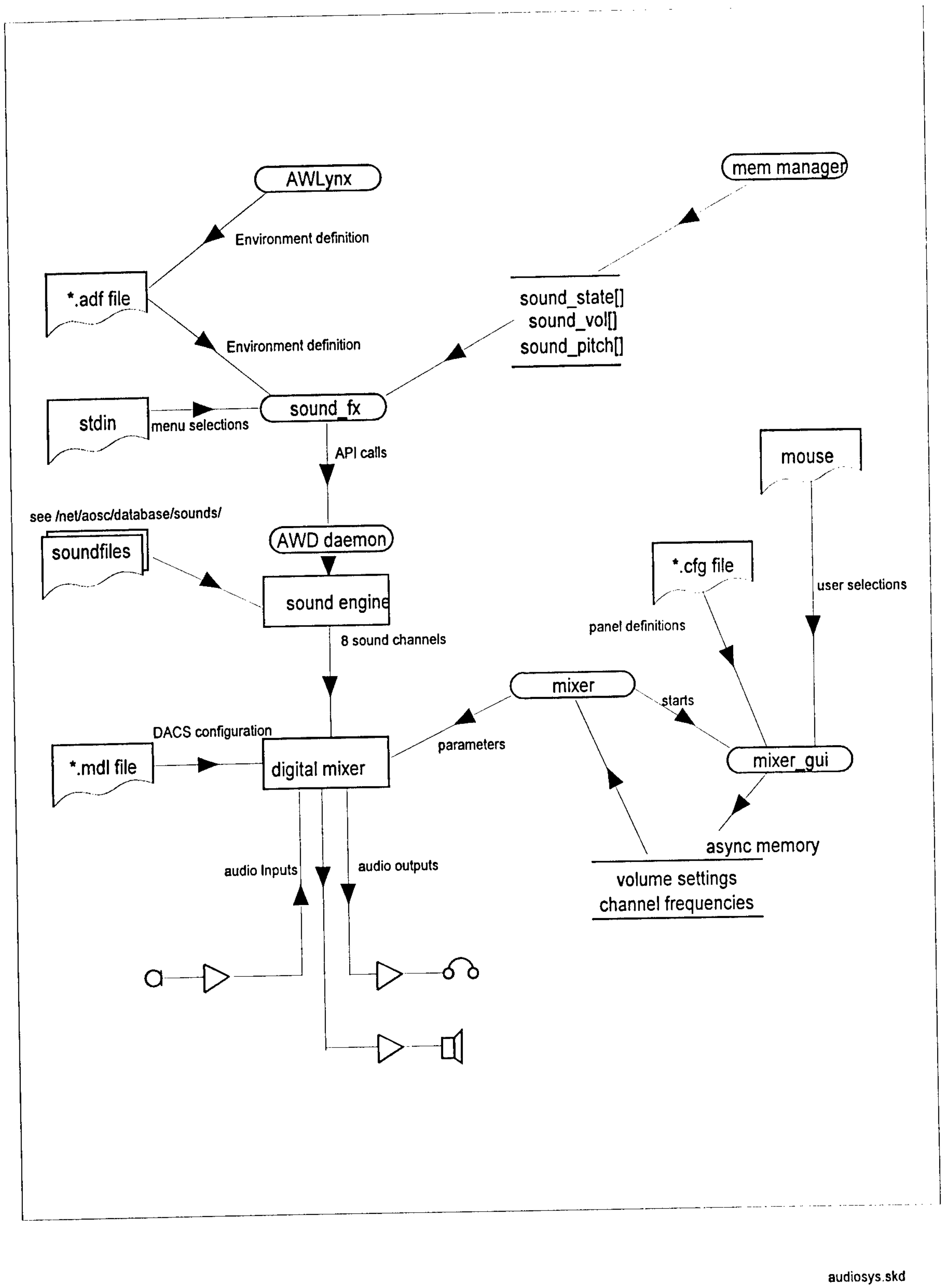

Figure 2 Context diagram of AOSC Audio System.

A general purpose configuration file named *'aosc.adf'* has been produced as a starting point for AOSC users. This file defines fixed and rotary wing *players*, each with their own pilot earpoints and associated soundfiles. This file can be used as-is or as a starting point when creating new configurations.

The *AWLynx* editor has been modified to enable the entry of additional parameters required by the general-purpose *sound_fx* program which controls sounds for AOSC applications. A *Sound Usage* flag has been provided in the *AOSC Controls* frame of the *AWLynx Sound Panel* to disable unwanted sounds defined in a given ADF configuration file. When sounds are enabled, a *Sound Index* frame is created so that sounds can be assigned to control variables, and an *Active on Startup* frame is available so that sounds can be automatically played from startup. The *Enable Retrigger* feature enables the user to specify whether the selected sound is to be treated as a one-shot edge-triggered sound which plays to completion once triggered, or a level-sensitive sound which restarts and plays only while its control variable is non-zero. See Appendix B, *'Guide to the AWLynx Editor'* for more details on the use of these parameters.

Whether using two or four channels to create a sound environment, *AW2* crudely adjusts sound volumes to create the image of a moving source. In AOSC applications, *AW2* must obtain the position of sound emitters from the corresponding entity positions defined in the tactical environment in order to simulate the moving sources. To provide users with an easy means of linking sound-emitting *players* to AOSC entities, an *Entity Id* field inside an *AOSC Reference* frame has been added to the *AWLynx Player* panel. The *sound_fx* program, discussed below, uses an entry in this field to search the tactical environment for the position of an entity and to transfer this data to *AW2*.

4.2 The AOSC *sound_fx* Program

The AOSC program *'sound_fx'* was created using *AW2* API functions to provide an easy method of controlling sound effects from within AOSC applications. The operation of the *sound_fx* program and usage of a modified *AWLynx* editor has been designed to eliminate the need for code modifications in most new applications.

By using the AOSC *memory manager* mechanism to transfer data from individual processes within the AOSC program environment, multiple programs can generate variables that control either the state, volume, or pitch of a sound within the simulated environment. The *sound_fx* program controls sound production via the *AW2 AWD* daemon by processing copies of these variables mapped by the *memory manager* mechanism.

Apart from modifying a standard ADF file and the usual *memory manager* mapping files, no code modifications will generally be required. The *sound_fx* program reads the ADF configuration file to obtain lists of required sound files and of corresponding

index numbers which assign each sound file to elements of the state-control, pitch-control and volume-control arrays. The *memory manager* mapping files define the origin of the external variables that are copied to the control array elements.

A menu displayed by the *sound_fx* program upon startup lists the options available to the user. By using the menu options, the user can list all the available sounds and the corresponding state, volume, and pitch values. Individual sounds can then be turned on or off by overriding their control variables using a few simple keystrokes.

See Appendix C *'Guide to the sound_fx program'* for details in the use of this program.

4.3 The AOSC *mixer* and *mixer_gui* Programs

Experiment operators can control parameters of the audio configuration loaded into the DACS digital mixer by using the *mixer_gui* program on any X-station. This program provides operators with adjustable slide controls selected on panels created from pull-down menus. A text-based configuration file loaded on startup defines labelled sliders which are functionally grouped into individual panels. Default slider settings can be specified in the configuration files so that the desired control parameter values can be loaded without requiring operator adjustment. Updates, if required, can be made by using a text editor to change the configuration files, or by adjusting slider controls and then selecting the *save* function.

The radio frequencies of modelled radios used by simulation participants can also be set or overridden by this program. Short lists of frequencies available in the simulated radios are specified in the configuration file. These available frequencies are listed against each radio displayed in a panel on the control console screen. The experiment operator may select from these lists to replace the frequency used by each radio. The first frequency listed against each radio also functions as the initial default frequency. By selecting the default frequency, external processes may override this value by mapping new updates via the AOSC *memory manager* mechanism. This feature enables experiment subjects to select their own radio frequencies when permitted.

The parameters controlled by the sliders and the selected radio frequencies are assembled into an asynchronous shared memory block to provide other processes with access to this data. The *mixer* program regularly transmits this data as a 256-byte UDP via an ethernet local area network to the DACS.

The parameters sent to the DACS can also be updated with variables generated in other processes. By using the *memory manager* mechanism, variables from other processes can be mapped into one of several small arrays created for external data. The configuration file entries determine where these external data array elements will be transferred within the UDP.

4.4 Standard SGI Audio Programs

The following SGI applications are available and may be of interest to audio users.

- The GUI-based *apanel* program controls the configuration of the audio port on the SGI machine in use. It is used to select the audio input source and to set sampling rates and record/playback volumes.
- The GUI-based *soundeditor* program records sounds from the selected audio port inputs and provides basic sound editing features. It creates AIFF files only.
- The GUI-based *soundfiler* program translates sound files to alternative formats.
- The *playaifc* program plays both AIFC and AIFF format files through the SGI audio ports.
- The *recordaifc* program records AIFC files from the selected audio port input.

For application developers, the SGI System Developers software includes the *AL* and *AF* audio libraries providing alternative functions for creating audio applications that operate the SGI audio ports. Special applications where the E-mu Systems Inc. EIIIxp sound engine is unsuited, such as some 3D sound experiments which may access over 1000 soundfiles, require coding using these libraries.

5. The Audio Distribution System

5.1 Equipment Rack and Signal Distribution

The major equipment components of the Audio Distribution System are arranged in a 19-inch rack located in the AOSC Computer Room. This rack contains the E-mu Systems Inc. EIIIxp sound engine, the ASTi DACS, and the multi-channel speaker and headphone amplifiers. The Convolvotron has been located adjacent to the rack enclosure since it cannot be rack-mounted. Connection panels to all equipment, and breakout termination panels for signal distribution cables are also mounted in the audio rack.

Patch cables are used for interconnections between the DACS equipment, the amplifiers, and the signal distribution cables.

Cables containing either four or six shielded circuits are used to distribute balanced line-level signals from the central rack to the areas requiring sound or voice communications facilities. These cables are located beneath a false floor so they are generally not visible unless floor tiles are raised. The areas serviced by the cables include the simulation bays, the Partial Dome Display and Helmet Mounted Display rooms, and the Control and Workstation Rooms.

Although it may be desirable to maintain all line-level[2] signals within the audio system at some standard level, this is difficult to achieve due to the variety of signal levels encountered in AOSC audio equipment. Since the DACS inputs and outputs include gain and attenuation stages designed to accommodate various signal levels, dissimilar signals from the consumer and professional equipment used in the Audio System can be easily interconnected provided the DACS is properly configured.

Various line signal levels exist within the audio system: The E-mu Systems Inc. EIIIxp sound engine and Australian Monitor amplifier operate at +4 dBu input levels, audio input levels to the video cassette recorders are at -10 dBu, the Convolvotron input/output levels are at 0 dBu, and microphone preamplifier outputs will vary depending upon microphone sensitivities but are expected in the range -10 to 0 dBu.

5.2 Microphones

Dynamic and condenser microphones are the most common microphone types in use today. The dynamic microphone is rugged, reliable, and requires no power to operate. In contrast, the condenser microphone having a very high impedance requires an integrated amplifier and must be powered by a 9 to 48 volt polarising supply. The electret condenser microphone does not require a polarising supply, but since it also contains an amplifier it requires either a 1.5 to 9 volt dc supply or a similar bias level applied to its output.

The general use of dynamic microphones throughout the AOSC audio system ensures that microphones are readily interchangeable and eliminates the problems of

---

[2] Although several standard signal levels are defined for audio circuits, signal levels are usually classified into one of three general categories:

- ***Mic or Low Level*** defines signals from sources such as microphones, cartridges, pickups and tape heads prior to any amplification and generally cover amplitudes less than -20 dBu (77 mV_{RMS}).
- ***Line or Medium Level*** defines signal levels at the input or output of signal processing equipment such as mixing consoles, equalisers, limiters, and recording equipment. It includes the outputs of preamplifiers and synthesisers, and the inputs to power amplifiers. These signals vary within the range of -20 dBu to +30 dBu (77 mV_{RMS} to 24 V_{RMS}).
- ***Speaker or High Level*** covers all signal levels above +30 dBu (>24 V_{RMS}). These levels are produced at power amplifier outputs.

The nominal line-level signals encountered in audio equipment are commonly at -16 dBu to -10 dBu (120-250 mV_{RMS}) levels in consumer equipment, 0dBu (0.77 V_{RMS}) or +4 dBu (1.2V_{RMS}) in professional audio equipment, and +8 dBu (2 V_{RMS}) in Compact Disc equipment.

The unit dBu or dbv is used to specify voltages in audio systems without regard for impedance. 0 dBu (or 0 dBv) corresponds to 0.7745 V_{RMS} . The unit dBm is used to specify power where 0 dBm corresponds to 1 mW. When appled to 600-Ohm loads, the scales are equivalent and 0 dBm corresponds to 0 dBu. Another common unit is the dBV which is referenced to 1 V_{RMS}.

supplying power to condenser-type microphones. Unfortunately electret condenser microphones have been supplied with the Helmet Mounted Displays, so these must use the modified microphone amplifiers inside the specially marked audio-breakout boxes.

Typical dynamic microphones have sensitivities within the range -80 to -70 dBu / 74 dB_{SPL}[3]. With normal speech at 0.3 metres (84 dB_{SPL}) and preamplifier gains of 60 dB, output levels from -10 to 0 dBu could be expected depending upon microphone selection. If standard line levels are required, these levels can be either trimmed to a standard -10 dBu or amplified to +4 dBu within the DACS.

5.3 Microphone Circuits

Balanced microphone circuits are used throughout the audio system so circuits are wired using shielded twisted-pair conductors. Shielding types within the installed cables vary but generally foil shields are used in fixed and underfloor wiring whereas braided or swerved shields are used in the flexible wiring accessible to users. Although foil shields provide nearly 100% shielding effectiveness, the shield may be damaged by frequent flexing so these are best used in fixed applications.

Microphone cables must be carefully placed away from conductors carrying large currents to minimise induced noise. Power and speaker cables should particularly be avoided, but when necessary cross these conductors at right angles to minimise pick-up.

Microphone signals are amplified near the microphones and transmitted to the audio rack through fixed wiring at line levels to minimise noise pick-up. The microphone amplifiers have balanced inputs designed for dynamic microphones and provide active balanced outputs. The inputs are protected against excessive voltages and contain bias circuits for use with electret microphones[4] if required.

5.4 Audio Connectors.

Interconnections within the audio system are achieved wherever possible through the use of standard 3-pin XLR connectors. These are widely used in professional audio systems because their rugged construction makes them most suitable for applications where they will be frequently unplugged, dropped, stepped upon, and otherwise

[3] The unit dB_{SPL} is used to specify sound pressure levels. 0 dB_{SPL} corresponds to a sound pressure level of 20 micro-Pascals which is the approximate hearing threshold of a young undamaged ear at frequencies of 1 kiloHertz. Microphone outputs levels are usually specified against a reference sound pressure level of 74 dB_{SPL} (1 micro-bar) which is the level of average speech at a distance of one metre.

[4] When modifying an amplifier input to provide a bias voltage for electret microphones, the amplifier gain should be reduced by 30 dB since these microphones produce much larger output signals.

abused. XLR connectors have large contact areas and a wiping contact-mating action that reduces contact resistance and helps maintain reliable connections.

The audio convention that the male contact supplies the signal has been adopted for low and line-level signals within the audio system.

The 6.5mm Tip-Ring-Sleeve (T-R-S) jack connector is used as the standard headphone connector at all user points. Because military headsets may use either T-R-R-S or T-R-R-R-S jack plugs, special adaptor cables have been produced to connect these to the standard headphone output socket.

For speaker applications, the Speakon connector has been used for all intermediate patches between amplifiers and speakers. This connector ensures reliable connections and features shrouded contacts to eliminate accidental shorts and minimise shock hazards. It also accommodates dual circuits so both left and right speaker channels may be connected in one action without the risk of cross-over connections.

At the user end, the speaker wiring is terminated into two separate AXR-PDN connectors for left and right channel outputs. These connectors provide reliable contacts and are designed to prevent inadvertent connections to the XLR-series connectors used elsewhere for line-level signals.

The IEC convention that the male pin carries the signal is not considered suitable at amplifier outputs. To minimise the risk of amplifier damage, the usual connector convention has not been adopted, but instead the alternative American usage of a female socket at all source outlets has been implemented.

The multicore cables, which distribute line-level microphone and headphone signals between the audio rack and all user stations, are terminated at the remote ends in 19-pin MIL-C-26482 circular connectors. The use of a single standardised connector ensures that all termination equipment or audio breakout boxes can be easily and quickly interchanged.

6. Future Developments

The AOSC Audio System has been developed to an operational stage capable of generating sound effects and providing audio communications between users. Although at this stage it generally meets the essential requirements of Reference 4, some additional hardware and software development is still required.

A multi-track computer-controlled audio recorder is still to be acquired for recording the audio communications generated by simulation participants. When a suitable recorder has been obtained, software for a user interface must be developed to provide operators at SGI workstations with record and playback control facilities.

Although the *mixer_gui* program enables an operator to control mixer parameters, future development of the Experiment Operator Station (EOS) may require extensions to the audio-control functions provided. At this stage no definition of requirements for these extensions exist but these may evolve as EOS development progresses.

7. Concluding Remarks

The development of the Audio System to date has provided the key components necessary to implement most of the audio functions required in the operation of the AOSC simulators. Sound effects, signal routing and communications channels required for routine AOSC operations are readily set-up by using the appropriate configuration files which have been created for these applications.

Audio functions supporting the AOSC F111 cockpit mock-up are well advanced. Aircraft noises are modelled and include engine turbine whine, afterburner roar, air-conditioning and wind noises, landing gear extension and retraction, and landing thumps and screeches. Simulated radio communications via three separate radios is available together with interphone communications between the pilot and navigator positions. Warning signals generated by cockpit instruments are developed as needed and have recently been produced for studies into development of more effective annunciators for the radar warning receivers. These functions form the basis for future developments or modifications supporting other cockpit mock-ups for fixed wing and rotary wing aircraft.

Communications between experiment operators and subjects is achieved through the use of simulated radios implemented within the DACS. These radios may be set by the experiment subjects to individual communication channels or overridden by the experiment operator for set-up or testing purposes. Experiment operators may also monitor discussions between subjects in dual-seat cockpits and record audio using stereo channels on a video cassette recorder.

The AOSC Audio System is now regularly used in the development and conduct of experiments. From this work, new requirements for additional sound effects or for new features in the basic system configuration have occasionally arisen and have been implemented without major effort.

8. References

1. R.A.Feik and W.M.Mason, 'The Use of Simulators to Improve Air Operations and Acquisitions in Australia', *SITRA 93*, Bandung, Indonesia, July 1993.

2. R.A.Feik and W.M.Mason, 'A New Simulation Facility for Research in Military Air Operations', *5th Australian Aeronautical Conference*, Melbourne, September 1993.

3. *'System Segment Specification for the Phase 1 Integration of the Air Operations Simulation Centre'*, Adacel Pty Ltd, 1994.

4. *'System Segment Design Document for the Phase 1 Integration of the Air Operations Simulation Centre'*, Adacel Pty Ltd, 1994.

5. *'EIIIx Emulator Three Reference Manual'*, E-mu Systems Inc., 1992.

6. *'Digital Audio System Documentation, Reference Manual'* Advanced Simulation Technology inc.

9. Bibliography

1. G. Davies & R. Jones, *'The Sound Reinforcement Handbook'*, Second Edition, Yamaha.

2. I. R. Sinclair, *'Audio & Hi-Fi Handbook'*, Second Edition, Newnes.

Appendix A - Table of Hardware Resources

Table A1 - The hardware resources available within the AOSC Audio System.

Location or Equipment	Available Resources
Audio Rack - Room G19	• E-mu Systems EIIIxp Sound Engine- 8 output channels. • Audio Amplifier - 6 x 30 W. • Australian Monitor Amplifier - 2 x 280 W. • Rane HC-6 Headphone Amplifier - 6 stereo channels. • Onyx Audio Port: 1 stereo headphone output, 1 stereo line-level input, 1 stereo line-level output, 1 digital input/output. • ASTi DACS Digital Mixer 48 sampled input & output channels.
Partial Dome Display - Room G17	• 6-core signal cable providing: (19-way connector) 2 microphone circuits, 2 headphone circuits, 2 spare circuits. • 2 x 2-channel speaker cables. (Speakon connector)
Helmet Mounted Display - Room G16B Experiment Operator Station - Room G16A Workstation Room - Room G20 Simulation Bay 2 - Room G15	• 6-core signal cable providing: (19-way connector) 2 microphone circuits, 2 headphone circuits, 2 spare circuits. • 2-channel speaker cable. (Speakon connector)
Simulation Bay 1 - Room G15	• 4-core signal cable providing: (19-way connector) 2 microphone circuits, 2 headphone circuits. • 2-channel speaker cable. (Speakon connector)
Black Audio Breakout Boxes, F18 cockpit, F111 cockpit, Blackhawk cockpit.	• 2 x dynamic microphone amplifiers. • Breakout connectors: 2 x XLR (F) for microphone input, 2 x 6.5mm T-R-S Jack Sockets for headphones, 2 x XLR (M) for spare circuits, 2 x AXR-PDN connectors for L/R speakers.

Appendix B - Guide to the AWLynx Editor

The AWlynx Editor

The use of the *AWLynx* editor is an important step in defining the configuration of a new sound-simulation application. It is anticipated that the easiest method of creating new applications will be by modifying existing ADF files with the *AWLynx* editor, so the file *'aosc.adf'* has been created as a basic AOSC configuration for this purpose.

The *AW2* software is installed in the directory *'usr/local/PSI/bin'* on both the AOSC fileserver and sound-engine host[5]. If this directory is included in your environment pathname variable, type *awlynx* from your current directory to start the editor.

Full descriptions of all *AWLynx* panels are available in the *AW2 Lynx User's Guide* but the reader may have difficulties identifying the relevant information from the many details presented. It is important for the reader who has no previous knowledge of the *AWLynx* editor but who must create an ADF file for a new application to distinguish between insignificant parameters which can remain in default states and those which must be carefully selected. The following sections briefly describe the more significant *AWLynx* panels and parameters which must be selected for AOSC applications. Default settings are provided without explanation for those parameters which need not be altered in most applications.

The *AWLynx* editor presents a GUI window displaying a set of icons along the left edge which each open a panel for defining the parameters for the corresponding class. To generate sounds, the details for the more important *AW Sound, AW Observer, AW Player, AW Channel* and *AW Scene* classes must first be defined. Each required sound must be listed in the *AW Sound* panel and referenced to the desired AIFF/AIFC wave files which will be down-loaded into the sound engine. The earpoints for simulation participants must be defined as *Observers* in the *AW Observer* panel, and sound-generating simulation entities are defined in the *AW Player* panel. The *AW Scene* panel is then used to place *Observers* into scenes together with the collection of sounds to be rendered into that scene. The channels used to output the sounds playing within a scene are also selected within the *AW Scene* panel.

[5] The AOSC fileserver is currently the SGI Origin *'Io'* and the sound-engine host is currently the SGI Onyx *'Elric'*.

The AW Channel Panel

All sounds heard at particular earpoints are grouped into *AW Scenes* and played through the corresponding channel which is defined in the *AW Channel* panel. Each channel comprises either two sound-engine outputs for stereo effects or four sound-engine outputs when all-round sound is required. Since the E-mu Systems Inc. EIIIxp sound engine has eight outputs, up to four stereo channels are available.

The **AW Channel** frame labels each required channel and the **Imaging Model** frame defines which audio outputs are allocated to that channel.

The E-mu Systems Inc. EIIIxp sound engine has sixteen voices which can each simultaneously replay a separate wavefile. The number of voices required by each channel is defined in the **Active Voices** frame, but careful selection is needed to ensure that no more than the total of sixteen voices is allocated across all channels.

Example settings are as follows:

• **AW Channel**	*PDD_channel*	Channel connected to PDD.
• **Engine**	*Default*	The sound engine to be used.
• **Image Model**	*Stereo*	Two channels required for headsets.
• **Left**	*Output1*	1 of 8 sound engine channels.
• **Right**	*Output2*	1 of 8 sound engine channels.
• **Active Voices**	*4*	12 voices remain for other channels.

The AW Engine Panel

The *AW Engine* panel defines the type and location of the sound generation facility required in a simulation. The audio port on any SGI machine can be specified as a sound source[6], although in AOSC applications the E-mu Systems Inc. EIIIxp sound engine is generally used. This panel requires no modification when using the E-mu Systems Inc. EIIIxp, and parameters should be set as follows:

• **AW Engine**	*Default*	Name of the selected engine.
• **Master Volume Control**	*1.00*	Set to full scale.
• **Headroom Control**	*6.0*	Allows 4 voices at max output prior to clipping.
• **AWD Specification**	*elric*	The sound engine attached to *Elric.*
• **Engine Specification**	*index*	*AW2* sorts out the details when *index* is used. (otherwise **/dev/emu/emu90d7** for EIIIxp)
• **Engine Index**	*1*	1 corresponds to EIIIxp sound engine, 0 corresponds to SGI audio port.

[6] An *AWD* daemon must be running on the specified machine prior to commencing *AW2* applications. This daemon will generally already be runing on the sound engine host machine.

- **Engine Access Mode** *shared* Default allows multiple processes.
- **Load Bank** *No* Use *yes* if wave files have previously been down-loaded into a sound bank stored on the sound engine internal hard disc drive.

The AW Env Panel:

This panel requires no modification. Parameters are set as follows:

- **Speed of Sound** *330* metres per second.

The AW Observer Panel

Each observer represents an earpoint that hears sound, and at least one observer must exist and be attached to a scene before any sound can be produced. Observers can be either stationary or moving. A stationary observer might represent an operator at a ground control station, whereas an aircraft pilot is an example of a moving observer.

Positions of stationary observers are defined in the **Coordinates** frame by selecting **Reference: Absolute** and inserting **X, Y, Z, H, P,** and **R** parameters in the relevant boxes.[7] In AOSC applications, these parameters are not updated so the observer position remains fixed.

Moving observers are defined by selecting **Reference: Player.** This selection causes observers to be tied to the player selected from the **player** frame. In AOSC applications, these player positions are updated by the AOSC *memory manager* mechanism from *location_north (N), location_east (E), and location_down (D)* entity coordinates defined in the tactical environment. Absolute or relative offsets from the selected player can be defined, but in most cases where the observer is a pilot within an aircraft these parameters can remain at the default zero values.

Note that the **Min** and **Max SPL** parameters that must be set correspond to the minimum and maximum volume settings for that channel. Sound levels set at or below the **Min** level will never be heard since the volume is effectively off. Alternatively, sound levels set at or above the **Max** level will be reproduced at levels corresponding to the maximum volume setting.

Example parameter settings are as follows:

[7] The AW2 X,Y,Z position coordinates correspond to AOSC coordinates *location_east (E)*, *location_north (N)*, and *-location_down (-D)* respectively.

• **Observer**	*F18_1_Pilot*	A moving observer.
• **Coordinates-Reference**	*Player*	Observer is tied to a player.
• **Coordinates-X,Y,Z,H,P,R**	*0*	Pilot is fixed inside aircraft.
• **Scene**	*F18_1_cockpit*	Sounds listed in this scene will be heard.
• **Channel**	*PDD_channel*	Sounds required in the PDD headsets/speakers.
• **Environment**	*Default*	No change necessary.
• **SPL-Min**	*0*	Threshold of hearing. 0 dB_{SPL} corresponds to minimum volume.
• **SPL-Max**	*120*	Maximum volume corresponds to 120 dB_{SPL}.
• **Default Modeling States**	*All*	Experiment here.

The AW Player Panel

Each dynamic sound-generating entity within the AOSC tactical environment must be assigned to an *AW2* player using the *AW Player* panel. Tanks, aircraft, missiles and ships are typical player examples which are created in the *AW Player* panel. As a player moves around the world, the active sounds attached to that player move to follow. All sounds that may be emitted by each player are listed in the **Sound Objects** frame. Each listed sound must be unique and not attached to any other player.

There are numerous options in defining the position of each player: player positions can be specified by individual world coordinates or by offsets relative to the positions of other players or observers. World coordinate player positions obtained from the tactical environment entity positions can be used in AOSC applications in which case **absolute** must be selected from the **Coordinate System** frame.

A special **Entity ID** frame has been created for AOSC applications so its description will not be found in the *AW2* manuals. A number must be entered which corresponds to an *Entity ID* number from the tactical environment. This entry enables the current player position to be updated from the corresponding entity positions defined in the tactical environment.

The **Player Position** frame defines initial positions for each player, but since these are immediately updated by the AOSC *memory manager* mechanism during simulations the default values can be used. Player positions are not updated if the **Entity ID** is zero so initial positions for stationary players are preserved in these instances.

Example parameter settings are as follows:

• **AW Player**	*F18_1*	Name need not match tactical environment entity name.
• **Sound Objects**	*F18_1_engine* *F18_1_missile_lock_tone* *F18_1_cockpit_background*	
• **Player Position-X,Y,Z,H,P,R**	*0*	Default values.
• **Coordinate System**	*Absolute*	World Coordinate positions used.
• **Entity ID**	*1*	Corresponds to tactical environment entity ID.

The AW Scene Panel

The sounds heard by individual observers within their environment are collected into groups defined in the *AW Scene* panels. Although the sounds associated with a particular aircraft are defined in a *AW Player* panel, the sounds heard by the pilot within the cockpit will differ from those heard by outside ground crew. The collection of sounds heard by these different *observers* are listed within separate *scenes* using the *AW Scene* Panel.

The **AW Scene** frame defines the set of scenes required for all observers, and the adjacent **Sound Objects** frame lists the sounds required within the selected scene. Example entries are as follows:

• **AW Scene**	*F18_1_cockpit*	Pilot environment within cockpit.
• **Sound Objects**	*F18_1_cockpit_background* *F18_1_missile_lock_tone* *Missile*	List of cockpit sounds

The AW Sound Panel

The *AW sound* panel is used to define all the sounds required in a simulation and to set the relevant parameters for each. When a new sound name is entered or an existing sound is selected in the **AW Sound** frame, the remaining panel frames become active for parameter entry.

The **File** frame is used to select the AIFF/AIFC wavefile from which the required sound will be created. Multiple sound objects referencing a particular wavefile may be defined.

The **Sound Pressure Level/SPL Fade (Lo)** and **Sound Pressure Level/SPL Fade (High)** frames assign sound pressure levels to the minimum and maximum volume limits for each sound. These levels are used to interpolate volume settings from sound pressure

levels specified within these ranges. However, in AOSC applications these entries are meaningless since sound levels are directly controlled by a volume variable. This variable may range between the limits 0 to 1.0 which directly correspond to the minimum and maximum volume settings respectively.

The **Pitch Bend/Pitch Bend Fade (Lo)** and **Pitch Bend Fade(High)** frames are provided for setting the minimum and maximum limits to the pitch bend ratio which can be applied to the selected sound. The pitch bend ratio may vary between a minimum of 0.667 and a maximum of 1.498. This range corresponds to a pitch variation of +/- 7 semitones. The required pitch-bend ratio is interpolated between the specified limits using a pitch variable within a range of 0 to 1.0. If this variable is unused it is set to the default value of 0.5 and the pitch-bend ratio is set halfway between the specified **Pitch Bend Fade** values.

The **Lopass Cutoff/Lopass Cutoff Fade(Lo)** and **Lopass Cutoff Fade(High)** frames are provided for setting minimum and maximum limits to the variable cutoff frequency of the low-pass filter which attenuates high frequencies in the selected sound. The low-pass cutoff frequency may be set from 0 to 22050 Hertz. The required cutoff frequency is interpolated between the specified limits using a lowpass-frequency variable with a range of 0 to 1.0. During AOSC applications, no mechanism is implemented to allow the cutoff frequency to be varied so the low-pass filter cutoff point will remain set at the value specified in the **Lopass Cutoff/Lopass Cutoff Fade(Lo)** frame.

The **Position** frame defines the world coordinates of a sound object not attached to a Player. When a sound object is attached to a player, this frames defines the offset of the sound object from the Player position.

The **Modelling States** frame is not important and default values can be used. Refer to the *AWLynx* manual for details of these parameters.

A special **AOSC Controls** frame has been included for parameters required in AOSC applications so its description will not be found in the *AW2* manuals. A number of parameters have been created here to simplify the interface required for controlling sounds.

The **Enable Retrigger** flag determines whether sounds are edge-triggered or level-sensitive. When *Enabled,* sounds are edge-triggered and are restarted on every OFF-to-ON transition of the corresponding state-control variable. Once triggered, a sound will play to completion unless again retriggered. When *Disabled,* sounds are level-sensitive and are only heard when the corresponding state-control variable is set to ON.

The **Active-on-Startup** flag controls whether sounds are automatically started when the *sound_fx* program is initialised. This feature is useful for starting continuous sounds such as cockpit background noises at simulation startups without the need for defining and mapping control variables.

When modifying existing ADF files, some previously defined sounds may not be required in the current AOSC application. To avoid losing previous programming effort in defining existing sound objects, it is undesirable to delete unused sound objects from an ADF file. The **Sound Usage** flag has been implemented to de-select those sounds which are not required in an application. Unwanted sounds with this flag set to *OFF* are ignored, whereas required sounds are selected by setting this flag to *ON*.

Each sound is controlled by separate state, pitch and volume variables. These variables are arranged into three arrays in which each sound is allocated the slot specified in the **Sound Index** frame. A **Sound Index** frame appears when a sound object is selected with the **Sound Usage** flag. The **Sound Index** entry must corresponds to the index selected in the relevant array of the mapping file used when importing external variables via the AOSC *memory manager* mechanism.

Although each sound must be assigned to a unique slot in the control-variable arrays listed in the mapping file, not all of the defined slots need be filled by the AOSC *memory manager* mechanism. Mapping may be omitted when sounds need not be controlled by external processes such as in instances when sounds are started automatically by the **Active-on-Startup** flag.

Appendix C - Guide to the *sound_fx* Program

The command line to initiate the *sound_fx* program is of the following form:

sound_fx **-f** *aosc.adf* **-r** *20* **-n** *soundmaker* **-s** *210* **-a** *-1*

The **-f** option specifies the AW2 configuration file for the current application, however *aosc.adf* is assumed if it is omitted. The remaining options are those required by the AOSC *memory manager* mechanism.

The *sound_fx* program may take some time to initialise if the ADF configuration file specified in the *-f* option requires new sound wavefiles to be down-loaded into the sound engine. Several minutes may elapse during this process.

When the program has been successfully initialised, a short menu appears which provides the user with several options:

- Enter **s** to examine a list of all available sounds and to view their current state and settings for volume and pitch.
- Enter **p** to view all defined players and to view their current position in world coordinates.
- Enter a digit from **0** to **9** to change the override settings for the required sound within the current bank. The override functions when enabled allow sounds to be switched permanently ON or OFF. Each time a number is pressed, the override-state of the corresponding sound advances through the cycle from *disable-override, override-ON,* to *override-OFF.* A message is printed each time a sound override-state is altered. Only sounds within the current bank can have their override settings altered. Advance to the next bank of ten sounds by pressing the *n* button.
- Enter **n** to advance to the next bank of ten sounds. This is useful only when changing the override states of individual sounds.
- Enter **q** to quit the program and kill all sounds.

Appendix D - Guide to the *mixer* and *mixer_gui* Programs

The parameters of the audio configuration loaded into the ASTi DACS can be controlled via any X-station using the *mixer* and *mixer_gui* programs. The *mixer* program creates shared memory areas from which control parameters are transmitted to the DACS via an ethernet local area network. Its companion program, *mixer_gui*, creates a graphical user interface that enables the user to make adjustments to the DACS control parameters transmitted via the shared memory. Both programs are essential and must be operating correctly to enable configuration tuning.

The *mixer* program is started by the command line:

mixer **-r** 10 **-a** 210 **-s** 220

The **-r** option defines the frequency at which this program executes and corresponds to the update rate of the DACS parameters. Slow update rates may be used when parameters are generated from only the *mixer_gui* program interfaces. However, faster parameter update rates may be required when the transmitted parameters include external signals generated at Experiment Subject Interfaces. For example, signals that indicate the operation of intercoms and radios require a quick response by the DACS and warrant higher update rates. An update rate of 10 per second, as shown in the above example, has been found to be the acceptable minimum when modelling radios controlled by experiment subjects.

The **-a** and **-s** options are essential and specify identification numbers for shared memory areas. The values shown above are examples only. If the *mixer* program cannot create or connect to the asynchronous shared memory denoted by the **-a** option, a warning message is displayed and the program aborts.

The *mixer_gui* program is started by the command line:

mixer_gui **-a** 210 **-f** config_filename

The **-a** option specifies the identification number for a shared memory area and must match that specified when starting the *mixer* program.

The **-f** option specifies the configuration text file which defines the parameters to be controlled. This file is essential but if it is not specified then *mixer.cfg* is assumed. A warning message is displayed and the program aborts if a configuration file cannot be loaded. The format of the configuration file is defined later.

The *mixer_gui* program reads the configuration file to create the entries in the *Comms_Control* panel which is immediately displayed. A sample panel is shown below.

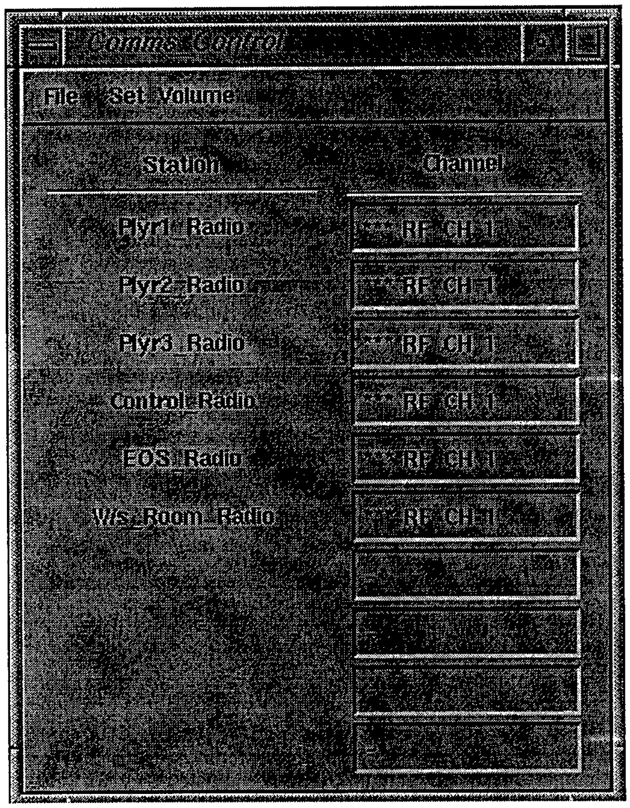

Figure D1 - Comms_Control Panel

Each radio or intercom station modelled by the DACS requires a channel parameter defining the operating frequency or channel number. At startup, the *Comms_Control* panel lists the current channel for each of the radio or intercom stations defined in the configuration file.

Up to five channels may be defined for each station and these can be selected to override channel selections made by the station operators. Use the right mouse button to view the available channels defined for each station or to make new channel selections. When the default channel is selected, as indicated by the *** prefix in the listings, the corresponding station operator selects the required channel via an external

variable supplied by the AOSC *memory manager* mechanism. In the absence of this external variable, the default channel is set to the first channel defined for each station.

If channel selections by remote station operators are required, external processes for the required operators must supply a channel-index variable within the range 1 to 5. The AOSC *memory manager* mechanism must then map these into the *station_ch* array created by the *mixer* program to effect the channel selection.

All channels included in station channel lists must be defined in the configuration file prior to their use.

Mixer gain or volume parameters may be adjusted by selecting the relevant control panel from the pull-down selection box under the *Set_Volume* menu item. The *SetVolume* control panels and the parameters they control are defined in the configuration file. A control panel, a sample of which is shown below, appears when a selection is made from the *Set_Volume* menu.

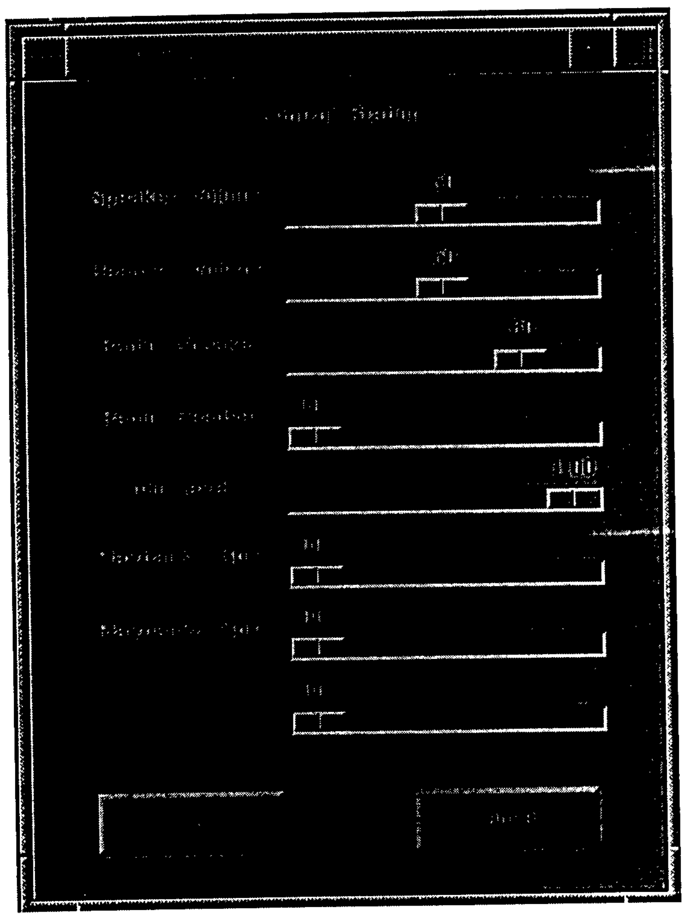

Figure D2 - SetVolume Control Panel.

Slider controls in the *SetVolume* control panels may be adjusted for immediate effects. When adjustments are complete, the panel may be closed by selecting the *OK* button to save all settings or by selecting the *Cancel* button to restore original settings.

To save new parameter settings for use again in later sessions, select *Save* from the *File* menu item in the *Comms_Control* panel. This action will write the new parameter values to the configuration file and copy the original file to config_filename.*cfg.old*.

Creating the Configuration File.

The configuration file required by the *mixer* program to define the contents of the panels shown above consists of a list of keywords and their associated arguments stored in simple text format. The keywords, their usage and limitations are detailed below.

Note that keywords must appear as the first item on a newline and in UPPERCASE text only. For many keywords, a numerical ***index*** value is specified which defines the destination of a parameter value within the UDP transmitted to the DACS.

The examples given in the following sections are actual lines used in creating the panels shown above.

 # this is an example of a comment line

Comment lines commence with the # character and can be located anywhere within the configuration file.

CHANNEL ***channel_name channel_frequency***

The CHANNEL keyword is used to define a channel that may be used on any radio or intercom station. For intercom channels *channel_frequency* may be any integer less than 100. For radio channels *channel_frequency* must be greater than 100 and given in Hertz.

Eg.	CHANNEL	RF_CH_1	27000
	CHANNEL	Service_Intercom	1

STATION ***station_name index channel_1 [.. channel_5]***

The STATION keyword is used to define each radio or intercom station and list the channels available for use by that station. Up to five channels may be listed but each channel must be defined prior to use. The *channel_frequency* corresponding to the selected *channel* is stored at the UDP destination *index*.

Eg. STATION Plyr1_Radio 50 RF_CH_1 RF_CH_2 Service_Intercom

PANELNAME panel_name
CONTROL control_name index **[initial_setting]**

The PANELNAME keyword creates a new *Set_Volume* panel for grouping the list of parameter CONTROLS which follow.

The CONTROL keyword assigns the next available slider control on the current *Set_Volume* panel to the parameter labelled *control_name*. The slider value is stored at the UDP destination *index*.

An optional *initial_setting* within the range 0 to 1.0 may be specified to preset sliders at startup.

Eg.	PANELNAME	Control_Station		
	CONTROL	Speaker_Volume	51	0.50
	CONTROL	Headset_Volume	52	0.50

DATA1 *index*

The DATA1 keyword is used to sequentially define the destination of each single-byte integer mapped into the ***ext_data1*** array created by the *mixer* program.

Eg.	DATA1	56	# ext_data1[0] is stored at location 56.
	DATA1	57	# ext_data1[1] is stored at location 57.

DATA2 *index*

The DATA2 keyword is used to sequentially define the destination of each 2-byte integer mapped into the ***ext_data2*** array created by the *mixer* program.

```
Eg.   DATA2   58      # ext_data2[ 0 ] is stored at location 58.
      DATA2   60      # ext_data2[ 1 ] is stored at location 60.
```

DATA4 *index*

The DATA4 keyword is used to sequentially define the destination of each 4-byte integer mapped into the ***ext_data4*** array created by the *mixer* program.

```
Eg.   DATA4   62      # ext_data4[ 0 ] is stored at location 62.
      DATA4   66      # ext_data4[ 1 ] is stored at location 66.
```

PRINT 0/1

The PRINT keyword is used to display each line as it is read from the configuration file.

```
Eg.   PRINT 1         # echo lines which follow to screen.
      PRINT 0         # stop line echo to screen.
```

LISTDATA

The LISTDATA keyword is used to list, after the configuration file has been processed, all external`data destinations defined with the DATA1, DATA2 or DATA4 keywords.

LISTCHANNELS

The LISTCHANNELS keyword is used to list, after the configuration file has been processed, all channels defined with the CHANNEL keyword.

LISTSTATIONS

The LISTSTATIONS keyword is used to list, after the configuration file has been processed, all stations defined with the STATION keyword.

LISTPANELS

The LISTPANELS keyword is used to list, after the configuration file has been processed, all *Set_Volume* panels defined with the PANELNAME keyword.

Appendix E - Troubleshooting AudioWorks2

Occasionally when running tested AudioWorks2 applications, no sound signals at all may emerge from the selected SGI audio port or E-mu Systems Inc. EIIIxp sound engine.

When AW2 suddenly stops working try the following procedure:

- Check to ensure that power is supplied to the E-mu Systems Inc. EIIIxp sound engine.

- Check that the AWD daemon is running by typing **ps -ef | grep awd**.
  If using the E-mu Systems Inc. EIIIxp sound engine, this daemon must be running on the sound-engine host.

- Type **/usr/local/PSI/bin/ awreset** and try the application again.

- Type **/usr/local/PSI/bin/ awinfo.**

  Awinfo should return a full description of the attached audio port and sound engine configurations listing all allocated channels, voices, and all stored sound wavefiles. Examine the listing to assess the states and availability of any attached audio port or sound engine. Proceed to the next step if unsatisfactory listings are obtained.

- Restart the AW2 daemon *AWD*.

  1. Using **root** or **superuser** privileges type **killall awd**.

  2. Check that all *AWD* processes have been removed by typing **ps -ef | grep awd.**

  3. Check that the sound engine is connected to its host by typing:
     /usr/local/PSI/bin/listscsi /dev/emu/*

     This command lists all the connected SCSI devices found. The following two entries must exist:
     a) the internal sound-engine hard disc drive entry:

        SCSI device **/dev/emu/emu90d4** inquiry info:
        vendor identification = MAXTOR
        product identificaton = 7213BSCSI
        product rev level = 1560

b) the sound engine interface entry:

SCSI device **/dev/emu/emu90d7** inquiry info:
vendor identification = E-mu Em
product identification = ulator III
product rev level = 0.10

Note that a bug in the current AW2 version (V1.3) prevents the sound engine interface entries from appearing while the *AWD* daemon is running.

4. Restart AWD by typing **/usr/local/ PSI/bin/awd.**
   Wait 10 seconds before typing **ps -ef | grep awd** to confirm that *AWD* has restarted.

Note that only one instance of *AWD* will be running if neither a sound engine nor an audio port is attached. An additional instance of *AWD* will be running if the E-mu Systems Inc. EIIIxp sound engine is attached, and up to two additional instances of *AWD* will be running if an audio port is attached.

DISTRIBUTION LIST

The Air Operations Simulation Centre Audio System

Reg Worl

1. DEFENCE ORGANISATION

S&T Program
Chief Defence Scientist
FAS Science Policy
AS Science Industry and External Relations
AS Science Corporate Management
(shared copy)
Director-General Science Policy Development
Counsellor Defence Science, London (Doc Data Sheet)
Counsellor Defence Science, Washington (Doc Data Sheet)
Scientific Adviser to MRDC Thailand (Doc Data Sheet)
Director General Scientific Advisers and Trials/Scientific Adviser Policy and Command (shared copy)
Navy Scientific Adviser (Doc Data Sheet and the distribution list)
Scientific Adviser - Army (Doc Data Sheet and distribution list only)
Air Force Scientific Adviser
Director Trials

Aeronautical and Maritime Research Laboratory
Director

Chief of Aircraft Operations Division
Research Leader Simulation Technology and Human Factors
Head Simulation Technology
Head Human Factors
Task Manager: J. Harvey
S. Parker
R. Martin
K. McNally
Author(s): R. Worl (2 copies)

DSTO Library
Library Fishermens Bend
Library Maribyrnong
Library DSTOS (2 copies)
Australian Archives
Library, MOD, Pyrmont (Doc Data sheet)

Capability Development Division
Director General Maritime Development, (Doc Data Sheet only)
Director General Land Development (Land),(Doc Data Sheet only)
Director General C3I Development (Air), (Doc Data Sheet only)

Army
ABCA Office, G-1-34, Russell Offices, Canberra (4 copies)

Intelligence Program
DGSTA, Defence Intelligence Organisation

Corporate Support Program (libraries)
OIC TRS, Defence Central Library
Officer in Charge, Document Exchange Centre (DEC), 1 copy
DEC requires the following copies of public release reports to meet exchange agreements under their management:
*US Defence Technical Information Centre, 2 copies
*UK Defence Research Information Center, 2 copies
*Canada Defence Scientific Information Service, 1 copy
*NZ Defence Information Centre, 1 copy
National Library of Australia, 1 copy

2. **UNIVERSITIES AND COLLEGES**

Australian Defence Force Academy
Library
Head of Aerospace and Mechanical Engineering
Senior Librarian, Hargrave Library, Monash University
Librarian, Flinders University

3. **OTHER ORGANISATIONS**

NASA (Canberra)
AGPS

OUTSIDE AUSTRALIA

4. **ABSTRACTING AND INFORMATION ORGANISATIONS**
INSPEC: Acquisitions Section Institution of Electrical Engineers
Library, Chemical Abstracts Reference Service
Engineering Societies Library, US
American Society for Metals
Documents Librarian, The Center for Research Libraries, US

5. **INFORMATION EXCHANGE AGREEMENT PARTNERS**
Acquisitions Unit, Science Reference and Information Service, UK
Library - Exchange Desk, National Institute of Standards and Technology, US

SPARES (7 copies)

Total number of copies: 55

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA	1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE	3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)
The Air Operations Simulation Centre Audio System	Document (U) Title (U) Abstract (U)
4. AUTHOR(S)	5. CORPORATE AUTHOR
Reg Worl	Aeronautical and Maritime Research Laboratory PO Box 4331 Melbourne Vic 3001

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-GD-0172	AR-010-512	General Document	April 1998

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
M1/9/341	RDI 95/180		37	6

13. DOWNGRADING/DELIMITING INSTRUCTIONS	14. RELEASE AUTHORITY
To be reviewed three years after date of publication	Chief, Air Operations Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE CENTRE, DIS NETWORK OFFICE, DEPT OF DEFENCE, CAMPBELL PARK OFFICES, CANBERRA ACT 2600

16. DELIBERATE ANNOUNCEMENT

No limitations

17. CASUAL ANNOUNCEMENT Yes

18. DEFTEST DESCRIPTORS

Flight simulation; Communications and radio systems; Sound systems; Sound engineering; Audio equipment

19. ABSTRACT

The Air Operations Division at the Aeronautical and Maritime Research Laboratory has established the Air Operations Simulation Centre for conduct of manned aircraft flight simulations. An audio system has been developed to deliver sound effects during simulations and to provide audio communications between manned stations. This report describes the hardware and software components of the audio system.

GENERAL DOCUMENT DSTO-GD-0172 AR-010-512 APRIL 1998

DSTO
AUSTRALIA

AERONAUTICAL AND MARITIME RESEARCH LABORATORY
PO BOX 4331 MELBOURNE VICTORIA 3001
AUSTRALIA, TELEPHONE (03) 9626 7000